# A METHOD AND SYSTEM FOR MODIFYING PRINT STREAM DATA TO ALLOW PRINTING OVER A SINGLE I/O PORT

5

## Related Applications

10      Reference is made to Application Serial Number 09/119,463, entitled A METHOD AND SYSTEM OF DISPLAYING DATABASE CONTENTS IN ENVELOPE DATA FIELDS, assigned to the assignee of this application and filed on July 20, 1998.

        Reference is made to Application Serial Number 09/119,183, entitled A METHOD
15  AND SYSTEM OF PRINT STREAM ADDRESS EXTRACTION, assigned to the assignee of this application and filed on July 20, 1998.

        Reference is made to Application Serial Number 09/119,464, entitled A METHOD AND SYSTEM OF PRINTING A POSTAGE INDICIA FROM AN ENVELOPE DESIGN
20  APPLICATION, assigned to the assignee of this application and filed on July 20, 1998.

        Reference is made to Application Serial Number 09/119,462, entitled A METHOD AND SYSTEM FOR CAPTURING DESTINATION ADDRESSES FROM LABEL DATA, assigned to the assignee of this application and filed on July 20, 1998.

25

        Reference is made to Application Serial Number 09/461,575, entitled a SYSTEM FOR ADDING SOFT FONTS TO A PRINTER DATA STREAM, assigned to the assignee of this application and filed on December 15, 1999.
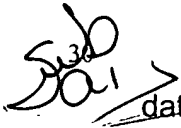
30

## Field of the Invention

The present invention relates generally to the field of print stream data processing; and, more specifically, to the field of printing to multiple device drivers via a single print stream input.

## Background of the Invention

' Mail preparation systems, such as the DOCUMATCH™ mail processing and finishing system available from Pitney Bowes Inc. of Stamford, Connecticut, establish a mail piece print run at a host personal computer (PC) and then direct the stream to printer peripherals for printing to an envelope and/or to a page substrate. Mail preparation systems are an example of systems whose purpose is to utilize address lists, perform addressing hygiene through the use of address correction techniques, assign barcoding and, download data to addressing printers, collators, sealers, and the like, for the purpose of producing a mailpiece.
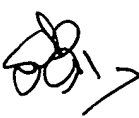
These systems sometimes have only a single input/output (I/O) port interface between the PC and the document printer. Thus, the current mail preparation systems are generally constrained by their printer hardware architecture.

To support such a system architecture, the application print data stream must be altered to allow the generation of envelope print data streams and its re-injection into the main application print stream. The creation of the envelope print stream involves the capture of text data contained in the document to generate the envelope, the use of an envelope definition module, and the use of proprietary print protocol language for the mail preparation system to direct the data to an appropriate printer.

The print stream created by the main application is generally in the form of text data, though it may take on other forms. The data must be parsed and checked before

2

~~format correction and barcoding techniques can be directed to the addresses~~ in the text
~~for creation of a mailpiece.~~

Mailpiece production systems are known in the art and have developed with
changes in postal service regulations (such as those of the United States Postal
Service, or USPS) and with the proliferation of appropriate software applications. In
turn, this production has served the need to automate and accelerate to accommodate
growth.

As the United States Postal Service (USPS), together with the postal services of
other countries around the world, moves toward more fully automated mail handling in
an effort to contain costs while processing ever increasing volumes of mail, automated
equipment which sorts and processes mail on the basis of machine readable postal
codes, such as the "zip code" or other forms of postal coding, play an ever more
significant role. In the United States, postal service regulations provide for a "Postnet"
bar code which represents the five, nine, or eleven digit zip code of the destination
address in a machine readable form. 4-State can be utilized within Canada.

Additionally, a system for printing envelopes with addresses including bar code is
disclosed in commonly assigned U.S. Patent No. 5,175,691 for a SYSTEM AND
METHOD FOR CONTROLLING AN APPARATUS TO PRODUCE ITEMS IN
SELECTED CONFIGURATIONS; issued on December 29, 1992 to Baker et al.
(hereinafter referred to as *Baker*), which describes a system for printing mail pieces
which includes a printer for printing sheets and envelope forms and a folder-sealer
mechanism for folding the envelope form around the sheets to form a mail piece, and a
computer based control system for controlling the printer and folder. In the system of
this application, when an operator is creating a file of letters to be printed, the operator
may designate a selected field within each letter as containing the destination address.
The system will then extract the information in this designated field and with it create a

new page of material to be printed on the envelope form; and, if the address within the designated field includes a zip code, the system will add a corresponding barcode to the new page. The system then adds this new page to the file before the file is output.

5    The ability to structure software coding is extremely important when linking data to be downloaded to a printer being utilized in the addressing environment. U.S. Patent No. 5,583,970 for a PRINTER COMMAND SET FOR CONTROLLING ADDRESS AND POSTAL CODE PRINTING FUNCTIONS, issued December 10, 1996 to Strobel (hereinafter referred to as *Strobel*), and assigned to the assignee of the present

10    claimed invention, is instructive in this respect.

*Strobel* is a method and system for printing images to a substrate wherein the commands normally input by an operator, or resident within the printer, can be determined at a host data processor. The system can control address and postal code printing functions beginning at the host computer together. The system will derive

15    printing data, including address data, from a selected application resident in the host computer. The host computer creates and then transmits printer command sets and printing data, via transmitting means to a microprocessor within the printer. The microprocessor drives a language interpreter which directs the printer commands to a parsing step for determining the address location from within the data to be printed. The

20    language interpreter then assigns delivery point digits to a zip code that was isolated from the transmitted address data. The newly created zip code is then matched with the bar code data stored within the microprocessor's corresponding memory. A bar code corresponding to the new zip code is selected. The language interpreter then directs the printer's controller to prepare to print the address with its corresponding zip code, any

25    graphics images that may have been included within the print data, and text, if any. The printer controller positions the bar code for printing, and then prints the bar code and address data, zip code, and any graphics images and text to an envelope or other substrate.

4

A particular limitation to current methods and systems, however, is found in the assembly of the envelope print stream which fuels the prior art detailed above. Mailpiece production systems having two separate printers and a single I/O port are constrained by the serial connection between their printers, making it impossible for the document printer to query information directly from its corresponding envelope printer. Therefore, as is illustrated by FIG. 2 hereinbelow, these mailpiece preparation systems must integrate two drivers while exposing only one to the application.

Therefore, it is an object of the present invention to provide for a method and system for determining and extracting an address from a print stream. Additionally, it is an object of the present invention to generate new print streams by printing to a secondary driver, reading the secondary print streams and injecting the secondary print stream back into the primary application print stream associated with a single I/O port connecting the mail preparation system to its host PC.

## Summary of the Invention

The invention is a method and system of modifying print stream data in a printing system having at least two printers and a single input/output port and comprises a number of steps and components.

The method begins by sending a print stream from a data processing application through a graphical device interface (GDI) to a print spooler to form a GDI print stream. The GDI print stream may contain: control data with a corresponding control page wizard which is utilized to facilitate mail merge functionality within the printing system; text data; address data; and/or, some other components. The data processing application can be a mailpiece designer application for preparing a mailpiece based on assigned parameters. Additionally, the mailpiece designer application is capable of

5

presenting a data entry screen to a system user for performing the further steps of creating and/or modifying a mailpiece definition file and, storing and/or retrieving one or more mailpiece definition files wherein each of the files corresponds to a specific mail print run. In a preferred embodiment, the document designer application is a 32-bit WINDOWS automation server.

The printing system employs a print stream monitor within a document driver kernel context for scanning the GDI print stream to determine whether or not the print stream comprises a set of text data and/or a set of address data. If the print stream comprises text data then the text data is tagged and sent to a user mode module; however, if the print stream does not comprise text data, then the print stream is sent directly to a data injection step. After tagging, the text data is stored in a local buffer. The tagged text stored in the local buffer cannot be retrieved until the stored tagged text has received an end of page control mark for the text sought to be retrieved.

The tagged and stored text data is then retrieved from the local buffer and it is determined as to whether or not an address is contained within the tagged text. The determination is made by an envelope parser for detecting, parsing, and then extracting address data from the print stream. If an address is found in the tagged text, then the address is placed in an envelope print format to create an envelope data set; however, if an address is not found, then the tagged text is sent directly to the data injection step. An envelope printer device context is then created and the envelope data set is transmitted to an envelope kernel for creating an envelope printer device language file.

The GDI print stream is converted by a document printer command language (PCL) generator into an envelope printer language. The envelope data resulting is then utilized by a second designer application for displaying a set of data fields of the envelope data to a system user, reading a set of parameters created by the second

designer application; and, writing the envelope data to a printer driver. The envelope data set is then printed.

Upon printing the envelope data set, the printer device language is then read by the print stream monitor which is used to modify the print stream by taking the envelope data set and injecting it back into the print stream from which it was extracted by merging the set of text data and the set of envelope data. The print stream is then transmitted to a next destination such as a document printer where a control page parser for detecting, parsing, and then extracting the document data from the print stream is employed. A printer command language (PCL) generator for converting the print stream into a document printer language is then employed. A printer driver is then activated for causing a printer to print the document data to one or more sheets.

## Brief Description of the Drawings

FIG. 1 is a block diagram of a system within which the method of the present invention could reside and be utilized.

FIG. 2 is a flowchart of the prior art method of printing envelope data extracted from a print stream in a WINDOWS® 95 environment.

FIG. 3A is a flowchart of the method of the present invention showing the initiation of the print stream and including a document driver kernel.

FIG. 3B is a continuation of the flowchart of the method of the present invention beginning with the document driver user mode module and concluding with the re-injection of data into the print stream.

FIG. 4 is a block diagram of the system of the present invention.

## Detailed Description of the Preferred Embodiments

5      The system concept and architecture describes the method and system required to print a mail piece run as defined by a user from a WINDOWS® application document and/or an address list to system printer. The printing application can be, but is not limited to, a word processor application and a mail list management or database application. Consequently, the system requirements include: the interception of the print 10   stream data from the documents being printed; parsing and extraction of certain information from the print stream data; adding information into the print stream data to generate a finished mail piece from the document being printed; and, converting the graphical device interface (GDI) print stream into a printer language print stream.

15      The following detailed description of the preferred embodiments presents the system requirements in context.

     Turning to **FIG. 1**, there is shown a block diagram of a system within which the method of the present invention could reside and be utilized.

20

     System **10** comprises a microprocessor **12** interoperatively connected to monitor **14** for viewing the representation of the medium (such as an envelope or label) to be acted upon by the design application **22**. The viewing of the media representation on monitor **14** promotes ease of use in selecting the various options available to the 25   system user while formatting the medium, and provides an example of the human interface that can be brought to system **10**. The monitor **14**, under control of the design application **22**, is able to show the system user: the medium representation; available menus from which option selections may be made; the medium's indicia; the amount of postage that will be incorporated into the indicia; and varied print fields available for

printing to the selected medium. Microprocessor **12** is interoperatively connected to scanner **16**. Scanner **16** provides system **10** with the ability to scan address field data, barcodes, or other scannable data sources as an input to design application **22**. Printer **26** is also interoperatively connected to microprocessor **12** and serves as the output

5    device by which the print fields are printed to the selected medium. Additionally, keyboard **20** is interoperatively connected to microprocessor **12** and serves as an input device for the input of data. Modem **18** gives system **10** the ability to communicate with other systems via communications means of varied types or to download print fields for remote storage; and, memory **24** allows the system to retain data for use in maintaining

10    records or for storing data for future use.

Turning to **FIG. 2**, there is shown a flowchart of the prior art method of printing envelope data extracted from a print stream in a WINDOWS 95 environment.

15    The prior art method begins at step **100** where a data processing application such as a mailpiece preparation application, operating in a WINDOWS 95 environment, initiates a print stream for each printed document. From step **100**, the method advances through a graphical device interface (GDI) at step **102** before entering a document driver module that begins at step **104**.

20

At step **104**, the method queries as to whether or not text data has been detected within the print stream. If the response to the query is "NO," then the method proceeds directly to step **118** where the data is re-injected into the print stream before advancing to step **120** where the sequence ends while the print stream is directed toward another

25    peripheral device. If the response to the query at step **104** is "YES," however, then the method advances to step **106** where the text data is stored in a local buffer to await an "end-of-page" control mark from the system.

At step **108**, the method queries as to whether or not an end-of-page control mark has been received at the local buffer. If the response to the query is "NO," then the method proceeds directly to step **118** where the data is re-injected into the print stream before advancing to step **120** where the sequence ends while the print stream is directed toward another peripheral device. If the response to the query at step **108** is "YES," however, then the method advances to step **110** where the data is retrieved from the local buffer before advancing to the query at step **112**.

At step **112**, the method queries as to whether or not an address has been found in the retrieved text. If the response to the query is "NO," then the method proceeds directly to step **118** where the data is re-injected into the print stream before advancing to step **120** where the sequence ends while the print stream is directed toward another peripheral device. If the response to the query at step **112** is "YES," however, then the method advances essentially simultaneously to steps **114** and **116**. At step **114**, the address data is printed to an envelope as envelope data, while at step **116**, an escape sequence is created using built-in printer command language (PCL) commands. Steps **114** and **116** rejoin at step **118** where the data is re-injected into the print stream before advancing to step **120** where the sequence ends while the print stream is directed toward another peripheral device such as a document printer.

Next turning to **FIG. 3A**, there is shown a flowchart of the method of the present invention showing the initiation of the print stream and including a document driver kernel.

The method begins at step **200** where a data processing application such as a mailpiece preparation application, operating in a WINDOWS NT environment, initiates a print stream for each printed document. From step **200**, the method advances through a graphical device interface (GDI) at step **202** and sends the print stream to a print

spooler at step **204** before entering document driver kernel **205** that begins with a query at step **206**. The GDI enables the system printer driver by initiating the query.

At step **206**, the method queries as to whether or not text data has been detected within the print stream. If the response to the query is "NO," then the method proceeds to step **208** where the data is sent to the user mode module, via **path A** to be re-injected into the print stream at step **236** as is shown in FIG. 3B. If the response to the query at step **206** is "YES," however, then the method advances to step **210** where the text data is tagged and sent to the document driver user mode module **225**, along **path B** as is shown in FIG. 3B.

FIG. **3B** is a continuation of the flowchart of the method of the present invention beginning with the document driver user mode module **225** and concluding with the re-injection of data into the print stream at step **236**.

Turning first to step **212**, **path B** is shown re-entering the method flow and the tagged text data is stored in a local buffer. From step **212**, the method advances to the query at step **214**. At step **214**, the method queries as to whether or not an end-of-page control mark has been received at the local buffer. If the response to the query is "NO," then the method proceeds directly to step **236** where the data is re-injected into the print stream before advancing to step **238** where the sequence ends while the print stream is directed toward another peripheral device. If the response to the query at step **214** is "YES," however, then the method advances to step **216** where the data is retrieved from the local buffer before advancing to the query at step **218**.

At step **218**, the method queries as to whether or not an address has been found in the retrieved text. If the response to the query is "NO," then the method proceeds directly to step **236** where the data is re-injected into the print stream before advancing to step **238** where the sequence ends while the print stream is directed toward another

peripheral device. If the response to the query at step **218** is "YES," however, then the method advances essentially simultaneously to step **220**. At step **220**, the system creates an envelope printer device context before advancing to step **222**. At step **222**, the address data is printed to an envelope driver as envelope data.

5

~~From step **222**, the method advances essentially simultaneously to steps **224** and **228**. Step **224** exists as a separate envelope kernel and user mode where an envelope printer device language (PDL) is generated for filing before advancing to step **226** where the envelope print data is placed in a temporary file. The method advances from step **226** to step **234** where the envelope PDL data file is read and then injected at step **236** back into the print stream before advancing to step **238** where the sequence ends while the print stream is directed toward another peripheral device such as a document printer.~~

10

15    Turning back to step **228**, the method queries as to whether or not all the available envelope data for this particular document has been printed. If the response to the query is "NO," then the method returns to re-enter the method flow at step **222** so that the remaining data can be printed. If the response to the query at step **228** is "YES," however, then the method advances to step **230** where the envelope printing is

20    ended. The method then advances to a query at step **232**. At step **232**, the method queries as to whether or not the system is ready to inject the envelope data back into the print stream. If the response to the query is "NO," then the method returns to re-enter the method flow at step **232** and the query is repeated. If the response to the query at step **232** is "YES," however, then the method advances to step **234** where the

25    envelope PDL data file is read and then injected at step **236** back into the print stream before advancing to step **238** where the sequence ends while the print stream is directed toward another peripheral device such as a document printer.

FIG. 4 is a block diagram of the system and corresponding components of the present invention.

A microprocessor **310** is shown interoperatively connected to a document design application **312**, for preparing a document such as a mailpiece with its associated text insert, to be printed as document text and as envelope text. In a preferred embodiment of the present invention, the document designer application **312** is a 32-bit WINDOWS automation server. The document designer application is capable of creating and/or modifying a mailpiece definition file and storing and/or retrieving one or more mailpiece definition files wherein each of said files corresponds to a specific mail print run and results in a print stream. Also connected to microprocessor **310**, is an envelope design application **314**.

The envelope design application **314** is utilized for: displaying a set of data fields of the envelope text data portion to a system user; reading a set of parameters created by the envelope designer application; and, writing the envelope data to a printer driver. The set of data fields displayed is representative of the face of an envelope (comprising an indicia print field and an addressee print field), thus allowing the system user a convenient way to check on field selection and placement.

Additionally connected to microprocessor **310** is a print stream monitor **318** for: scanning the print stream generated by the mailpiece design application **312**; detecting a set of document data or control data and a set of envelope data; interfacing with the envelope parser **320** to extract an address from the document text data; interfacing with the document parser **322** to extract control page information from the print stream; generating the envelope PCL print data at the PCL print generator **324**; and, for modifying the print stream to merge the two sets of data. The print stream monitor **318** maintains the system timing during printing of the mailpiece and the general performance of the document print job. The

The system includes a document (or control page) parser **322** for detecting, parsing, and then extracting the document data from the print stream, as well as instructing the print stream task manager (not shown). Further included, is an envelope parser **320** for detecting, parsing, and then extracting the envelope data from the print stream and then indicating to the print stream task manager that an address has been detected.

To print the envelope, a PCL generator **324** is connected to the microprocessor **310** for converting the envelope data as extracted from the print stream into a second printer language, thus creating a proper PCL for the envelope text to be printed through printer driver **328** to printer **332** and on to an envelope or similar substrate. To print the document, a PCL generator **326** is connected to the microprocessor **310** for converting the document data as extracted from the print stream into a second printer language, thus creating a proper PCL for the document text to be printed through printer driver **330** to printer **334** and on to one or more sheets or similar substrate. The WINDOWS NT printer architecture requires that every printer driver be implemented as a pair of user mode dynamic link libraries (DLL), as well as a printer specific component.

The WINDOWS NT printing architecture is part of the NT graphics architecture and consists of three components; these are: the applications that interface with the WINDOWS GDI; the server print spooler that interfaces with the print services convention; and, the kernel mode print services that include the printer driver minidriver and, the I/O port interface. The print spooler internally accesses the user interface and the user mode printer driver components. The minidriver is a data file that contains the printer data tables as well as code specific to system driver that works in conjunction with the shim common driver. The purpose of the shim common driver is to intercept the raster graphic entry points to obtain print stream data and to provide each of the other drivers with common kernel functions when necessary.

14

While certain embodiments have been described above in terms of the system within which the method may reside, the invention is not limited to such a context.

5      In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

10